FLAGS SOFTWARE

INTERVIEW WEB APP

WHAT IS THIS?

By now you have read the readme and got the app started.

The whole task should be quick to perform to complete the hard requirements. There are clearly many avenues for improvements, which is nice if you want to show something off, but don't spend too much time on this.

This page is for information only. The underlying code for this page is not intended to be modified or reused, and once replaced with your own code, the containing folder can be safely deleted.

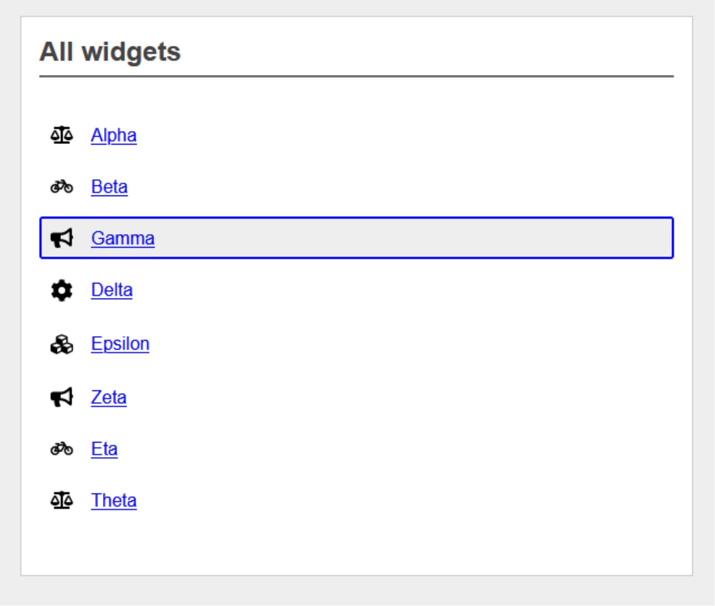
REQUIREMENTS

The following are hard requirements:

- Home page that fetches & displays a list of widgets
- Each widget should link to a Widget Detail page
- The Widget Detail page should fetch the details for the widget and display basic information on that widget
- Each API request may sometimes fail. This needs to be handled in this app
- The API is provided by miragejs (see `/src/api/api.ts`). The code providing it can be read, but should not be altered.
- The code should be made available in a public repository (github, gitlab, etc). This repo should include your commit history.

ROUGH UX

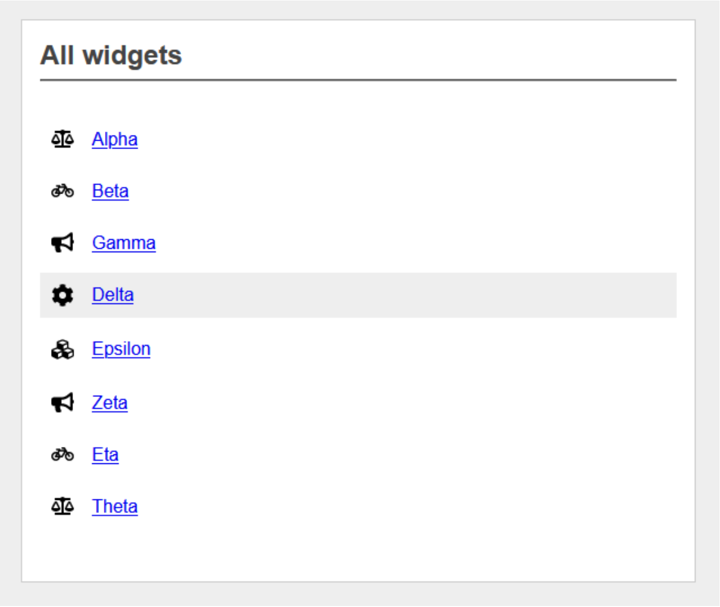
The following are outlines of the sort of information we want to see displayed. This is not a pixel-perfect requirement, but a broad stroke. The design is primarily to be simple, functional, readable and consistent, but there is scope for beautifying it should you wish to.



All widgets

- Alpha
- Beta
- Gamma
- Delta
- Epsilon
- Zeta
- Eta
- Theta

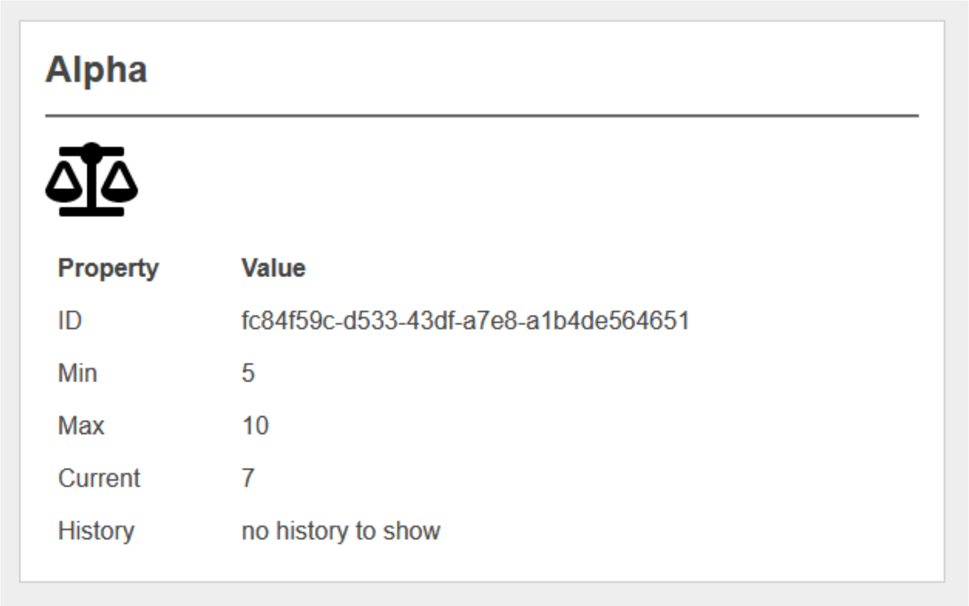
Widget list, including focus state.



All widgets

- Alpha
- Beta
- Gamma
- Delta
- Epsilon
- Zeta
- Eta
- Theta

Widget list, including mouse over hover state.



Alpha

Property	Value
ID	fc84f59c-d533-43df-a7e8-a1b4de564651
Min	5
Max	10
Current	7
History	no history to show

Widget detail. Should show the name in the title bar, the image and the properties shown.

ADDITIONAL INFORMATION

Only the basic hard requirements are laid out here, and should not take a long time to implement. There is, of course, a lot of scope to sell your skills in other areas than just being able to update a basic React App, should you wish to do so.

The technologies we use at present include, but are not limited to:

- state management using react-redux, redux-saga and redux-thunk (not necessarily all at once)
- styling using styled-components
- routing using react-router
- fetching API data using apisauce (a thin wrapper around axios)

This is not a list of required dependencies to use, if you have experience in other dependencies, feel free to use those instead. If you feel there is not the need, you do not need to use them.

WHAT ARE WE LOOKING FOR?

Primarily, we are looking for well-organised, consistent code that is easy to read and easy to extend. In addition:

- It should be standards compliant and accessible
- It should be robust, handle failure gracefully and should always inform the user what is happening (or has happened) as clearly as possible
- The code you provide should be easy to read, and easy to extend
- If a dependency is used, there should be a clear reason for using it
- The resulting app should look reasonable, but that is not the primary concern here
- Comments in code are fine where you consider them necessary, but are not required
- This project uses Typescript, but there are no hard requirements to use it
- There should not be any unused code remaining by the time you submit it to us
- Testing is nice but not required

You are free to use libraries to simplify your code or to enable additional functionality.

TECHNICAL DETAIL

- The widgets list API endpoint is at `GET /api/widgets`
- The widget detail API endpoint is at `GET /api/widgets/:id`
- The API endpoints do not consistently return successfully
- The data returned is not necessarily of the highest quality. Some properties may occasionally be missing.

TALKING POINTS

In addition to the requirements, the project would eventually require other features. Please consider the following features and give us an overview of how you would approach solving these issues

- The API requires authentication
- The widgets list provides more items than the front end can display at once
- The widgets list should be sortable
- The widgets list API request is too slow, so we want to filter the list down before fetching it
- The widget detail page needs to show a chart
- The page needs to be viewed on devices of varying sizes, from mobile devices to 4K screens
- We need to display a list of doodads and a details page
- We need to migrate to a mobile app
- We want the information displayed on the page to be up to date whenever the page is viewed. This is to be automatic and seamless.